VipulSunilPatil(22110189)@Ass3

Que1. Write a program in assembly language to subtract two 16 bit numbers without using the subtraction instruction. Note: the numbers have to be fetched from the memory.

```
.data
  # num1 is s0 and num2 is s1
  num1: .word 21 # 0x1234
  num2: .word 7 # 0x4321
  out: .word 0

.text
  # Load address of num1, num2 and out
  la $s0, num1
  la $s1, num2
  la $s2, out

  # Load the num1 and num2 from memory
  lw $t0, 0($s0)
  lw $t1, 0($s1)

  # 2's complement of num2
  not $t1, $t1
  addi $t1, $t1, 1

  # addition on complement of num2
  add  $t2, $t0, $t1

  # store output in out
  sw $t2, 0($s2)

  # printing out
  li $v0, 1
  lw $a0, 0($s2) # add $a0, $zero, $s2
  syscall
```

using only temporary variable

```
.data
   # num1 is s0 and num2 is s1
   num1: .word 21 # 0x1234
   num2: .word 7 # 0x4321
   out: .word 0
.text
   # Load address of num1, num2 and out
   la $t0, num1
   la $t1, num2
   la $t2, out
```

```
    # 2's complement of num2
    not $t1, $t1
    addi $t1, $t1, 1
    # addition on complement of num2
    add  $t2, $t0, $t1
    # store output in out
    sw $t2, out

    # printing out
    li $v0, 1
    lw $a0, $t2 # add $a0, $zero, $s2
    syscall
```

Que2. Write an assembly language program to find an average of 15 numbers stored at consecutive locations in memory.

```
.data
  # array is our list of numbers and len is length of total numbers ie, 15
  array: .word 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70, 75
  len: .word 15
  # avg is were we will be getting avrage of all numbers# avg is were we will be getting 
avrage of all numbers
  avg: .word 0

.text
  main:
    # Load address of num1, num2 and out
    la $t0, array
    li $t1, 0   # initial index in array
    lw $t2, len
    li $t3, 0  # initial sum is 0 (to store sum of number upto current index)

    loop:
      # Performing calcutations (sum = sum + array[i])
      lw $t4, ($t0)
      add $t3, $t3, $t4

      #Incrementing index and array  (ie, moving to next index of array)
      addi $t1, $t1, 1 # can also use add
      add $t0, $t0, 4

      #Check condition
      blt $t1, $t2, loop # $t1 < $t2

    # Calculating avrage (sum/length)
    div $t5, $t3, $t2

    # store output(avrage) in avg
    sw $t5, avg
    # printing avg
    li $v0, 1
    lw $a0, $t5 # avg
    syscall

    #Temiate
    li $v0, 10
    syscall
```

Que3. Write an assembly language program to find an LCM of two numbers stored at consecutive locations in memory.

```
.data
  # Store data of num1, num2 Consicative position
  array: .word 15, 5
  # lcm is were we will be getting lcm of num1 and num2
  lcm: .word 0

.text
  main:
    # Load address of num1, num2 at Consicative position
    la $t0, array
    lw $t1, ($t0)
    lw $t2, ($t0 + 4) # Consicative

    # Need to calculate GCD of 2 numbers (using Euclidean algorithm)
    gcd:
      beq  $t2, $zero, gcd_done # t1(rem) is equal to zero then gcd_done and t1 is gcd
      rem $t3, $t1, $t2  # t2(15) divide t1(5)

      move $t1, $t2      # t1 ko t2(5) banao ie, t1 = 5
      move $t2, $t3      # t2 ko t3(rem) banao ie, t2 = rem

      j gcd

    gcd_done:
      # Calculate LCM
      mul  $t5, $t1, $t2  # Multiply num1(15) and num2(5)
      div  $t5, $t5, $t4  # LCM (num1 * num2) / GCD(num1, num2) ie, t4

      # store output in lcm
      sw $t5, lcm

    # printing lcm
    li $v0, 1
    lw $a0, $t5 # lcm
    syscall

    #Temiate
    li $v0, 10
    syscall
```

Que4. Write an assembly language program to calculate multiplication of two numbers without using MUL commands.

```
.data
  # Storing data of num1 and num2
  num1: .word 7
  num2: .word 21
  # multi is where we will be finding final output (ie, num1 * num2)
  multi: .word 0

.text
  main:
    # Load address of num1, num2
    lw $t0, num1
    lw $t1, num2
    li $t2, 0     # initial loop countor (incimentor)
    li $t3, 0     # initial multiplication is 0

    loop:
      # Performing calcutations (t3 = t3 + num1(7))
      add $t3, $t3, $t0
      # Increment the loop counter
      addi $t2, $t2, 1

      #Check condition
      blt $t2, $t1, loop # t2 < num2(21)

    # store output in multi
    sw $t3, multi

    # printing multi
    li $v0, 1
    lw $a0, $t3 # multi
    syscall

    #Temiate
    li $v0, 10
    syscall
```

Que5. Write an assembly language program to find a given number in the list of 10 numbers (assuming the numbers are sorted). If found store 1 in output, else store 2 in output.The given number has been loaded from X location in memory, the output has to be stored at the next location and if found store the number of iterations and the index of the element at the next at the next consecutive locations, if found.

```
.data
  # list is our array, find is the number we have to find in array
  # and len is length of total string ie, 5
  list: .word 1, 2, 3, 4, 5, 6, 7, 8, 9, 10
  find: .word 7
  len: .word 10
  # findIndex is where we will be geting index of array where found number in
  findIndex: .word -1 # Assuming -1 when not find number (find) in array (list)

  # Asked in question
  output: .word 0 # Assuming not found
  iterations: .word 0 # Assuming no iterations done
  index: .word -1 # Assuming -1 when not find number (find) in array (list)

.text
  main:
    # Load address of str, find and len
    la $t0, list
    lw $t1, find
    li $t2, 0   # initial index is 0
    li $t3, len
    li $t4, 0   # initial iteration count

  loop:
     # Load the number at current index into $t5
    lw $t5, ($t0)
    # Compare character and if yes jump to found
    beq $t5, $t1, found
    # Increment the index counter
    addi $t2, $t2, 1 # add
    addi $t0, $t0, 4  # Move to next number
    blt $t2, $t3, loop # curr index < len

  notfound:
    # store output in findindex
    li $t6, 0
    sw $t6, output
    # as we are asked for iterations and index only if, found
    # sw $t2, iterations
    # sw $t2, index ie, it will be 10
    j exit

  found:
    # store output in findindex
    li $t6, 1
    sw $t6, output
    sw $t2, iterations
    sw $t2, index
    j exit

  exit:
    # Not asked in Question
    # # printing findindex
    # li $v0, 1
    # lw $a0, $t3 # findindex
    # syscall

    #Temiate
    li $v0, 10
    syscall
```

Que6. Write an assembly language program to find a character in a string.

```
.data
  # str is our string, find is the char we have to find in str
  # and len is length of total string ie, 5
  str: .asciiz "Vipul"
  find: .byte 'u'
  len: .word 5
  # findIndex is where we will be geting index of str where found char in
  findIndex: .word -1 # Assuming -1 when not find char (find) in sting (str)
.text
  main:
    # Load address of str, find and len
    la $t0, str
    lw $t1, find #lb
    lw $t2, len
    li $t3, 0      # initial index is 0
    li $t4, -1      # initial findindex is -1 same reson (as in findIndex)

    loop:
      # Load the character at current index into $t3
      lw $t5, ($t0) #lb
      # Compare character and if yes jump to found
      beq  $t3, $t5, found

      # Increment the index counter
      addi $t3, $t3, 1 #add

      # Check condition for notfound
      # bge $t1, $t2, notfound # If index >= len, jump to notfound
      # Check condition for loop
      blt $t3, $t2, loop # curr index < len

    notfound:
      # store output in findindex
      sw $t3, findindex
      j exit

    found:
      # store output in findindex
      sw $t3, findindex
      j exit
    exit:
      # printing findindex
      li $v0, 1
      lw $a0, $t3 # findindex
      syscall

      #Temiate
      li $v0, 10
      syscall
```

inside loop we need to add this
addi $t0, $t0, 1 # Move to next character

after addi $t3, $t3, 1 #add and before blt $t3, $t2, loop # curr index < len

Github Repo Link: https://github.com/vipulSP2108/ES-215-Computer-Organization-and-Architecture/tree/main/Assignment3